1:  
a. De ander zal nu worden in geladen omdat de ene nu uit de kritieke sectie is.  
b. Er kan een Starvation optreden als hij niet snel genoeg uit die kritiek sectie komt.

2:  
Ze zijn deelbaar omdat ze een collineaire verband kunnen delen.

3:  
Het ligt aan de begin waarde S. Als deze niet hoog genoeg is dan zal hij nooit bij de V aankomen omdat die al geleegd is.

4:  
a. Er wordt al met de waarde gedeeld zodat het andere gedeelte het gebruik er van kan krijgen.  
b. Er zullen fail safes komen voor als hij de if niet ingaat.

5:  
a. Bij lifo krijg je het probleem dat de eerste nooit aan de beurt komt en bij FIFO heb je kans dat de laatste er nooit uit komt.  
b. Zulke processen heten starvation-free, lockout-freed of ze moeten een finite bypass hebben.

6:  
De volgorde van hoe de process zijn op gebouwd.

7:  
buffer[in] = item; in = (in + 1) % BUFFER\_SIZE;

item = buffer[out]; out = (out + 1) % BUFFER\_SIZE;

Er wordt hier maar rekening gehouden met 1 item.

8:  
1

9:  
De data verwerking moet worden afgevangen. Hierna moet er meerdere fail safes komen om het fatsoenlijk te houden.

10:  
Het kan zijn dat hij het dom weg omwisselt maar het kan ook zijn dat hij erop vast loopt.

11:  
a.

Chef: while(true) { pak volgend memo; verwerk memo; }   
Ambtenaarl while(true) { chef(). }  
b.

Chef: while(true) { pak volgend memo; verwerk memo; Ambtenaarl().}   
Ambtenaarl while(true) { chef(). }  
c.

Chef: while(true) { pak volgend memo; verwerk memo; Ambtenaarl().}

Ambtenaarl while(true) { chef(). }